Assignment #2

An Exercise of SQL Using SQL*Plus(2)

KAIST

Myoung Ho Kim

Table of Contents

- **SQL*Plus Commands**
- **Assignment #2**
  - SQL
  - Relational algebra expression

SQL*Plus Commands

SQL*Plus Commands

- **SELECT ** FROM tab***
  - list all the tables stored in the database

TAB	Attribute Name	Type
	TName	VARCHAR2(30)
	TabType	VARCHAR2(7)
	ClusterID	NUMBER

```
SQL> create table TEST (
     2  name  char(10));
Table created.

SQL> create table TEST2 (
     2  name  char(10));
Table created.

SQL> create table TEST3 (
     2  name  char(10));
Table created.
```

```
SQL> select  tname  from tab ;
TNAME
----------------------------------------------
TEST
TEST2
TEST3
3 rows selected.
```

SQL*Plus Commands (cont'd)

- **DESC <table_name>**
  - shows information of a table <table_name>

```
SQL>create table STUDENT (
    2  studentId NUMBER PRIMARY KEY,
    3  name VARCHAR2(10),
    4  department VARCHAR2(25),
    5  semesters NUMBER,
    6  doubleMajor VARCHAR2(25));
Table created.
```

```
SQL> DESC STUDENT;
 Name                      Null?      Type
 ------------------------- ---------- ----------
 STUDENTID                 NOT NULL   NUMBER
 NAME                                 VARCHAR(10)
 DEPARTMENT                           VARCHAR(25)
 SEMESTER                             NUMBER
 DOUBLEMAJOR                          VARCHAR(25)
```

SQL*Plus Commands (cont'd)

- **Transaction commands**
  - ***ROLLBACK***
    - » Undo changes (transactional).
  - ***COMMIT***
    - » Save changes in database (transactional).
  - ***SET AUOTOCOMMIT ON***
    - » commit pending changes to the database
      after Oracle Database executes each successful INSERT, UPDATE.
  - ***SET AUTOCOMMIT OFF***
    - » suppress automatic committing so that you must commit changes manually.

SQL*Plus Commands (cont'd)

- **Transaction commands**
  - *rollback;*

```
SQL>insert into dept values(50, 'AD',
'SEOUL');
1 row created.

SQL>select * from dept;

DEPTNO  DNAME          LOC
------  -------------- --------------
    30  SALES          CHICAGO
    40  OPERATIONS     BOSTON
    50  AD             SEOUL
```

```
SQL> rollback;

SQL>select * from dept;

DEPTNO  DNAME          LOC
------  -------------- --------------
    30  SALES          CHICAGO
    40  OPERATIONS     BOSTON
```

Undo work done

SQL*Plus Commands (cont'd)

- **Transaction commands**
  - *commit;*

```
SQL>insert into dept values(50, 'AD',
'SEOUL');
1 row created.

SQL>select * from dept;

DEPTNO   DNAME          LOC
------------  --------------------  ---------------
      30   SALES          CHICAGO
      40   OPERATIONS  BOSTON
      50   AD                SEOUL
```

```
SQL> commit;
Commit complete.

SQL>select * from dept;

DEPTNO   DNAME          LOC
------------  --------------------  ---------------
      30   SALES          CHICAGO
      40   OPERATIONS  BOSTON
      50   AD                SEOUL

SQL> rollback;
Rollback complete.

SQL>select * from dept;
 DEPTNO   DNAME          LOC
 ------------  --------------------  ---------------
      30   SALES          CHICAGO
      40   OPERATIONS  BOSTON
      50   AD                SEOUL
```

Save changes

Outerjoin Syntax

- **Right outer join**

(Ex) MovieStar(name, address, gender, birthdate)
MovieExec(name, address, cert#, netWorth)

```
– MovieStar NATURAL RIGHT OUTER JOIN MovieExec;
```

MovieStar

name	address	gender	birthdate
Mary T. Moore	Maple St.	'F'	9/9/99
Tom Hanks	Cherry Ln.	'M'	8/8/88

MovieExec

name	address	cert#	networth
Mary T. Moore	Maple St.	12345	$100…
George Lucas	Oak Rd.	23456	$200…

Result

name	address	gender	birthdate	cert#	networth
Mary T. Moore	Maple St.	'F'	9/9/99	12345	$100…
George Lucas	Oak Rd.	NULL	NULL	23456	$200…

Outerjoin Syntax (cont'd)

– MovieStar NATURAL RIGHT OUTER JOIN MovieExec;

1.
```
SELECT *
FROM MovieStar NATURAL RIGHT OUTER JOIN MovieExec
```

2.
```
SELECT *
FROM MovieStar star RIGHT OUTER JOIN MovieExec exec
ON star.name = exec.name AND
      star.address = exec.address;
```

3.
```
SELECT *
FROM MovieStar star, MovieExec exec
WHERE star.name(+) = exec.name AND
      star.address(+) = exec.address;
```

Assignment #2

Submission

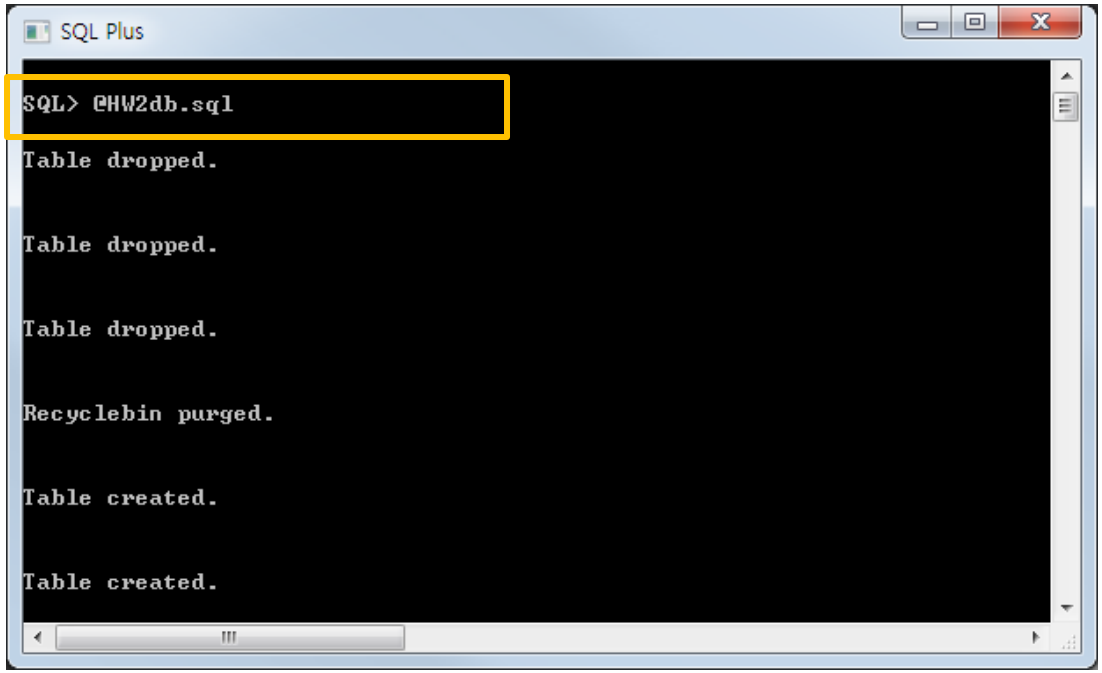
- **Due**
  - March. 30, 12 p.m.
  - Delay is not accepted
- **Submission standard**
  - *[student ID].lst* contains the executions of SQL commands and their results. You may use ***SPOOL*** command.
  - *[student ID].docx* includes relational algebra. For given queries, write them in relational algebra expression.
    - » Write your student ID at first line of the file
  - Archive them into *[student ID].zip* and upload it to course homepage
- **Evaluation**
  - You will get points if your SQL queries use the assigned operators and find the right answers.
  - You will get points if your Relational algebra find the right answers.
  - Do not cheat others. Both of them will get no point.

Example Database

- **Create tables for homework.**

  1) Download *HW2db.sql* from the course homepage and Copy it to (directory that *Oracle Client* is installed)₩BIN

  2) ***@HW2db.sql*** or ***start HW2db.sql***



Example Database (cont'd)

- **Database Design**
  - You can see all the tables stored in your database using a command '**select * from tab**'

EMPLOYEE	EmployeeID	Name	Department	Salary	Manager
	70547	Rachel	Research	5000	
	80201	Mike	Development	4900	YES
	…				

WORKS_ON	EmployeeID	ProjectID	Hours
	70547	SP204	33.0
	80201	SP322	31.0
	…		

PROJECT	ProjectID	Name	Location	Start Date
	P101	ProductX	Houston	2012-08-25
	P202	ProductY	New York	2011-12-17
	…			

❖ EMPLOYEE.Manager : If the employee is a manager, then this tuple has value 'YES' (String value)

Queries

- **Q1. Aggregation and Group By**
  - For each project, list *projectID*, *location* and *the sum of working hours* of the employees who work on the project.
    Order the output by *location* in lexicographic order. If locations are the same, list the project first which has less employees.

Queries (cont'd)

- **Q2. Having**
  - Find the projects which satisfy all the conditions below
    - Location is 'Chicago'
    - The number of employees who work on the project is more than 3
    - The average salary of the employees of the project is more than 4200

    List the *ID* of these projects and the *sum of working hours* of the employees who work on the project.

Queries (cont'd)

- **Q3. CREATE TABLE**
  - **Create** a table that contains the information of the *department.*
    - » The table has two attributes,
      - the name of the department
      - the number of employees who work on the department

DEPARTMENT	Name	NumOfEmployee
	Headquarter	3
	Research	7
	...	

Queries (cont'd)

- **Q4. INSERT**
  - **Using one SQL statement, insert** all the tuples to the DEPARTMENT table which you made in Q3
    - » You can insert tuples using SELECT-FROM-WHERE clause
    - » After inserting, execute 'SELECT * FROM department' command.

Queries (cont'd)

- **Q5. UPDATE (Use 'IN')**
  - For each employee who works on the project whose location is 'LA', increase his/her salary by 5%.
    - » After increasing, list *name* and *salary* of every employee.

Queries (cont'd)

- **Q6. OUTER JOIN**
  - For each department, list the name of the department and the manager name. For departments that do not have manager, just list its name.

Relational Algebra Expression

Write each query as relational algebra expression on [*student ID*].docx

- **Q-a**. For each project, list *projectID*, *location* and *the sum of working hours* of the employees who work on the project. Order the output by *location* in lexicographic order. Among same locations, list the project first which has less employees.

- **Q-b**. For each project whose location is 'Chicago' and *the number of employees* is more than 3 and *average salary* of employees are more than 4200, list the *projectID* of the project and *sum of working hours* of the employees who work on the project.

- **Q-c**. For each employee who works on the project which location is 'LA', increase his/her salary 5%.

- **Q-d**. For each project, list *name* and *salary* whose salary is higher than all of the employee of the project (Without duplication)

References

- **Lecture notes**
- **Text book**
  - 2.3, 6.4, 6.5
- **Oracle Tutorial**
  - http://www.holowczak.com/oracle/sqlplus/